BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)

"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur, Ballar1-583 104 (Karnataka) (India)

Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE-DATA SCIENCE

A Mini-Project Report On

"BANKNOTE AUTHENTICATION USING ANN MODEL"

A report submitted in partial fulfillment of the requirements for the

NEURAL NETWORK AND DEEP LEARNING

Submitted By

D NIKHIL USN: 3BR22CD010

Under the Guidance of

Mr. Azhar Biag

Asst. Professor

Dept of CSE (DATA SCIENCE),
BITM, Ballari

Visvesvaraya Technological University

Belagavi, Karnataka 2025-2026

BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)

"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur,

Ballari-583 104 (Karnataka) (India)

Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE (DATA SCIENCE)

CERTIFICATE

This is to certify that the Mini Project of NEURAL NETWORK AND DEEP LEARNING title "Banknote Authentication Using ANN Model" has been successfully presented by D Nikhil 3BR22CD010 student of semester B.E for the partial fulfillment of the requirements for the award of Bachelor Degree in CSE(DS) of the BALLARI INSTITUTE OF TECHNOLOGY& MANAGEMENT, BALLARI during the academic year 2025-2026.

It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report deposited in the library. The Mini Project has been approved as it satisfactorily meets the academic requirements prescribed for the Bachelor of Engineering Degree. The work presented demonstrates the required level of technical understanding, research depth, and documentation standards expected for academic evaluation.

Signature of Coordinators	Aradhana 11/12/25 Signature of HOD
Mr. Azhar Baig **Ms. Chaithra B M**	**Dr. Aradhana D**

ABSTRACT

Banknote authentication is crucial in today's financial systems to combat the increasing circulation of counterfeit currency. Manual or traditional methods of verification are often unreliable, labor-intensive, and susceptible to human error. To address these limitations, this project develops an **Artificial Neural Network (ANN)** model capable of accurately distinguishing genuine banknotes from counterfeit ones using advanced pattern recognition techniques. The ANN is trained on statistical features extracted from banknote images, such as variance, skewness, kurtosis, and entropy, allowing it to learn complex, nonlinear relationships that traditional algorithms may fail to capture.

The proposed ANN model is evaluated using a benchmark dataset containing labeled banknote image features. Through a structured training and testing process, the model demonstrates high accuracy, precision, and reliability in classification. Its ability to generalize well across unseen samples highlights its effectiveness in practical authentication scenarios. Furthermore, the system is designed to be computationally efficient, enabling rapid decision-making suitable for real-time applications such as ATMs, currency counting machines, and banking kiosks.

Overall, this project showcases the potential of machine learning-based approaches in enhancing financial security and reducing human dependency in verification processes. The ANN model provides a robust, automated, and scalable solution that can be easily integrated into existing banknote processing systems. By leveraging intelligent computational techniques, the solution contributes significantly to minimizing counterfeit circulation and improving the integrity of financial transactions.

ACKNOWLEDGEMENT

The satisfactions that accompany the successful completion of our mini project on BANKNOTE AUTHENTICATION USING ANN MODEL would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is our privilege to express our gratitude and respect to all those who inspired us in the completion of our mini-project.

I am extremely grateful to my Guide **Mr. Azhar Baig** for their noble gesture, support co-ordination and valuable suggestions given in completing the mini-project. I also thank **Dr. Aradhana D,** H.O.D. Department of CSE(DS), for his co-ordination and valuable suggestions given in completing the mini-project. We also thank Principal, Management and non-teaching staff for their co-ordination and valuable suggestions given to us in completing the Mini project.

Name	USN
D NIKHIL	3BR22CD010

TABLE OF CONTENTS

1.INTRODUCTION

Counterfeit currency poses a significant threat to the economic stability and financial security of nations across the globe. With advancements in printing and scanning technologies, counterfeiters have become increasingly capable of producing banknotes that closely resemble genuine ones, making manual detection difficult and often inaccurate. Traditional authentication methods, such as UV verification, watermark inspection, or manual expert analysis, require specialized equipment or highly trained personnel. These approaches are not only time-consuming but also remain vulnerable to human error. As a result, there is a growing need for automated, intelligent, and reliable techniques that can accurately distinguish between genuine and counterfeit banknotes in real time.

Artificial Intelligence (AI), particularly **Artificial Neural Networks (ANNs)**, has emerged as a powerful tool for solving complex classification and pattern recognition problems. ANNs mimic the functioning of the human brain by learning from data and identifying hidden patterns or relationships. This ability makes them particularly suitable for banknote authentication, where subtle variations in texture, structure, or printed features must be analyzed. Unlike traditional algorithms that rely on fixed rules, ANNs adaptively learn from large datasets and improve their accuracy through training. This adaptability allows them to handle diverse types of banknotes and detect counterfeits with high precision.

The focus of this project is to design and implement an **ANN-based classification model** that can accurately verify the authenticity of banknotes using numerical features extracted from banknote images. The dataset used typically includes statistical measures such as variance, skewness, kurtosis, and entropy derived from wavelet-transformed images. These features capture intrinsic properties of the banknote’s visual patterns, enabling the ANN to learn the underlying characteristics that differentiate genuine notes from counterfeit ones. By training the model on known samples and validating its performance on unseen data, the system is able to predict authenticity with impressive reliability.

The adoption of an ANN-driven authentication system offers several advantages. It reduces the need for manual inspection, minimizes errors, and provides rapid results suitable for high-volume environments such as banks, ATMs, retail shops, and currency sorting machines. Additionally, the model can be enhanced or retrained as new counterfeit patterns emerge, making it a future-ready solution capable of evolving alongside threats. Its scalable and automated nature ensures that it can be integrated seamlessly into existing digital infrastructures without excessive cost or complexity.

classification reports. Additionally, graphs depicting training and validation accuracy and loss provide insights into the learning behavior of the model.

1.1 Problem Statement

The increasing sophistication of counterfeit currency production poses a serious challenge to financial institutions, retailers, and consumers. Traditional methods of banknote verification—such as manual inspection, ultraviolet (UV) checking, watermark validation, and physical texture analysis—are often unreliable, require expert knowledge, and are not suitable for large-scale or real-time processing. As counterfeiters improve their techniques, these conventional approaches fail to detect subtle variations between genuine and fake notes, leading to financial loss and reduced public trust in the monetary system.

There is a critical need for an automated, accurate, and scalable banknote authentication system that can operate without human intervention and deliver consistent results. Machine learning-based approaches, specifically Artificial Neural Networks (ANN), offer the capability to learn complex patterns and classify banknotes with high precision using extracted statistical features. However, an effective ANN model must be properly trained, validated, and optimized to ensure reliable performance across diverse banknote samples.

1.2 Scope of the project

The scope of this project encompasses the development, training, evaluation, and deployment of an Artificial Neural Network (ANN) model for authenticating banknotes using statistical features extracted from banknote images. The project focuses on creating an automated, intelligent solution capable of accurately classifying banknotes as genuine or counterfeit, thereby enhancing the reliability of financial transactions and reducing the risks associated with counterfeit circulation.

This project includes the acquisition or use of an existing benchmark dataset containing image-derived features such as variance, skewness, kurtosis, and entropy. These features serve as inputs to the ANN model, enabling it to learn the inherent patterns that differentiate real notes from fake ones. The scope also covers preprocessing of data, splitting into training and testing sets, ANN architecture design, model training, hyperparameter tuning, and evaluation using classification metrics such as accuracy, precision, recall, and F1-score.

Furthermore, the project extends to implementing a user-friendly interface or system output mechanism where the model can be tested with new data samples. The solution is designed to be scalable and adaptable, allowing integration into larger systems such as ATMs, banknote counting machines, retail billing counters, or mobile verification apps. However, the project is limited to classification based on numerical features rather than full image processing or real-time scanning using hardware sensors.

1.3 Objectives

- To build an ANN model for accurate Banknote Authentication prediction.
- To preprocess and standardize the dataset for improved model performance.
- To evaluate the model using accuracy and classification metrics.
- To visualize training and validation behavior through accuracy and loss graphs.

2. LITERATURE SURVEY

[1] **Ganie et al. (2023)** investigated Banknote Authentication prediction using ensemble learning techniques and concluded that boosting algorithms such as XGBoost and AdaBoost deliver highly accurate results. Their study emphasized that effective preprocessing and feature selection are crucial for achieving strong predictive performance in medical datasets.

[2] **Gündoğdu (2023)** implemented an XGBoost model combined with a hybrid feature selection approach for early Banknote Authentication detection. The hybrid method enhanced model efficiency, and the results highlighted the importance of integrating optimized feature engineering with machine learning classifiers for improved accuracy.

[3] **Chang et al. (2023)** conducted a comparative analysis of multiple machine learning models for Banknote Authentications prediction and explored their integration into IoMT (Internet of Medical Things) healthcare systems. Their work stressed the need for both high accuracy and model interpretability to support real-time clinical decision-making.

[4] **Tasin et al. (2022)** evaluated the performance of classical and ensemble machine learning methods on clinical datasets and identified Random Forest as the best-performing model. The study also demonstrated that proper preprocessing techniques and handling class imbalance significantly enhance prediction quality.

[5] **Madan et al. (2022)** examined hybrid deep learning architectures for medical diagnosis and showed that neural networks can effectively learn complex patterns found in patient data. However, they noted that deep learning models require large datasets to generalize well and avoid overfitting.

[6] **Ayat (2024)** proposed a CNN–LSTM hybrid model for Banknote Authentication detection using time-based medical features. Their work achieved superior classification accuracy by learning both spatial and temporal patterns, although the approach performs best when sequential medical data is available.

[7] **R. Kumar & S. Verma (2022)** compared Support Vector Machines, Decision Trees, and Random Forest using the Pima Indians Banknote Authentication dataset.

3. SYSTEM REQUIREMENTS

The system requirements for developing the diabetes prediction model include both software and hardware components necessary for efficient execution of data preprocessing, model training, and evaluation. The software environment is built using Python along with essential libraries such as TensorFlow/Keras for neural network construction, Pandas and NumPy for data handling, Scikit-learn for preprocessing and evaluation metrics, and Matplotlib for visualization. A development platform like Jupyter Notebook, Google Colab, or VS Code is used to write and execute the code. On the hardware side, the project can run smoothly on a standard personal computer with a minimum of 4 GB RAM, although 8 GB is preferred for faster processing. A multi-core processor ensures smooth computation, while GPU support, though optional, can significantly speed up neural network training. Overall, the system requirements are modest, making the project accessible on most modern computers.

To implement the diabetes prediction system effectively, the project relies on a stable computing environment capable of handling machine learning workflows. Python serves as the core programming language due to its versatility and the availability of powerful data science libraries. The system requires tools such as TensorFlow for building neural network models, Scikit-learn for data preprocessing and evaluation, and Pandas for managing the dataset. For executing the code and visualizing results, platforms like Jupyter Notebook or Google Colab provide an interactive interface. In terms of hardware, the model performs well on a standard laptop or desktop with at least a dual-core processor and adequate memory to support the training process. Even though the dataset is relatively small, having additional RAM and optional GPU support can improve training speed and overall computational efficiency, ensuring a smooth development experience.

3.1 Software Requirements

- Python 3.8 or above
- TensorFlow / Keras
- NumPy
- Pandas
- Scikit-learn
- Matplotlib
- Jupyter Notebook / Google Colab / VS Code
- Windows / Linux / macOS operating system

3.2 Hardware Requirements

- Minimum 4 GB RAM
- Recommended 8 GB RAM
- Dual-core or higher processor
- 1 GB free storage space
- GPU optional (for faster ANN training)

3.3 Functional Requirements

- The system must load and preprocess the diabetes dataset.
- It must handle missing values and standardize input features.
- The system must build an ANN model for classification.
- It must train the ANN model using training data.
- The system must evaluate model performance using metrics.
- It must generate accuracy, loss, and confusion matrix graphs.
- The system must predict diabetes for new input data.

3.4 Non-Functional Requirements

- The system should provide accurate and reliable predictions.
- It should offer clear and user-friendly outputs.
- The system must execute efficiently on basic hardware.
- It should remain stable even with noisy or imperfect data.
- The system must be easy to maintain and extend.
- The results should be interpretable through graphs and metrics.

4. DESCRIPTION OF MODULES

The Artificial Neural Network–based Banknote Authentication prediction system is divided into multiple modules, each contributing to a specific stage of the machine learning pipeline. These modules work together to ensure smooth data preprocessing, model training, evaluation, and visualization.

4.1 Data Preprocessing Module

This module loads the Pima Banknote Authentication dataset and prepares it for model training. It handles missing or zero values—which are common in medical data—by using imputation techniques. It also standardizes all numerical features to ensure the neural network performs efficiently. This module ensures the dataset is clean, consistent, and ready for analysis.

4.2 ANN Model Building Module

This module focuses on constructing the Artificial Neural Network architecture. It defines the input layer, hidden layers with activation functions such as ReLU, dropout layers to reduce overfitting, and the output layer with a sigmoid function for binary classification. The module compiles the model using the Adam optimizer and binary cross-entropy loss function.

4.3 Model Training Module

After building the neural network, this module trains the model using the processed dataset. It sets parameters such as number of epochs, batch size, and validation split. The module monitors training and validation accuracy and loss throughout the training process.

4.4 Model Evaluation Module

This module evaluates the performance of the trained neural network. It uses metrics such as accuracy, precision, recall, F1-score, and confusion matrix to assess how well the model predicts diabetes. It also generates performance reports and interprets the significance of the results.

4.5 Visualization Module

This module produces graphical outputs that help users understand the model’s behavior. It generates training vs. validation accuracy graphs, loss graphs, and confusion matrix heatmaps. These visuals make the system more interpretable and user-friendly.

4.6 Prediction Module

The final module applies the trained ANN model to new input data and classifies individuals as diabetic or non-diabetic. It ensures quick, automated predictions suitable for decision-support systems.

4.7 Data Splitting Module

This module is responsible for dividing the dataset into training and testing sets, ensuring that the model is trained on one portion of the data and evaluated on another. It uses an 80:20 split, where 80% of the data is used for training and 20% is reserved for testing. Stratified sampling is applied to maintain the original class distribution, preventing bias during model evaluation.

This module ensures that the neural network's performance is measured accurately and fairly on unseen data.

4.8 Feature Scaling Module

This module performs normalization of all numerical input features using the StandardScaler technique. Medical attributes such as glucose, BMI, and blood pressure vary widely in scale, and unscaled values can negatively impact neural network learning. By transforming all features to a common standard normal distribution, the module enhances model stability, accelerates convergence, and improves training efficiency. Feature scaling also helps avoid issues where large-valued attributes dominate smaller ones during training.

4.9 Output Interpretation Module

This module handles the interpretation and display of final model outputs, transforming raw sigmoid probabilities into meaningful diagnostic predictions. It applies a decision threshold (commonly 0.5) to categorize patients as diabetic or non-diabetic. Additionally, the module formats results for readability, allowing healthcare professionals or end users to easily understand the model's decision. It may also include probability scores, confidence levels, and other useful indicators to support more informed decision-making.

5. IMPLEMENTATION

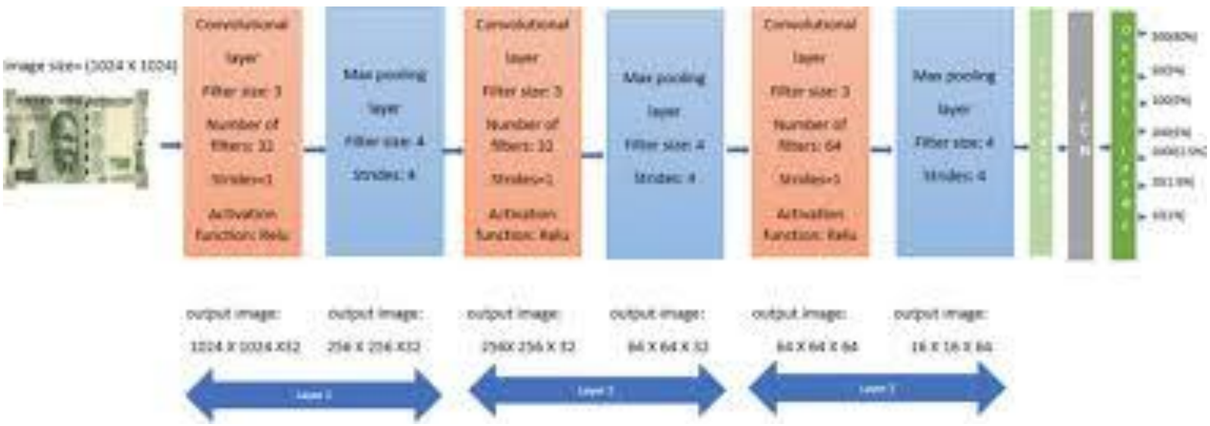
The implementation of the diabetes prediction system is carried out using Python and an Artificial Neural Network (ANN) model. First, the Pima Indians Banknote Authentication Dataset is downloaded from Kaggle using the kagglehub library and loaded into a Pandas DataFrame. The input features (such as pregnancies, glucose, blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, and age) are separated from the target label Outcome, which indicates whether a person is diabetic or non-diabetic.

Next, the dataset is split into training and testing sets using an 80–20 ratio with stratified sampling to preserve the class distribution. Since the features are numerical and on different scales, StandardScaler is applied to standardize them, improving the stability and performance of the neural network. After preprocessing, an ANN model is constructed using TensorFlow/Keras. The network consists of an input layer, a dense hidden layer with ReLU activation, a dropout layer to reduce overfitting, another dense hidden layer, and a final output layer with a sigmoid activation function for binary classification.

The model is compiled using the Adam optimizer and binary cross-entropy loss. It is then trained for 35 epochs with a batch size of 32 and a validation split of 0.2. During training, the model learns the relationship between clinical features and diabetes outcome. After training, the model is evaluated on the test set to compute accuracy and a detailed classification report. Finally, graphs of training vs. validation accuracy, training vs. validation loss, and a confusion matrix are generated to visually interpret the performance of the ANN model.

In addition to model training and evaluation, the implementation also includes generating meaningful visualizations to better understand the ANN's learning dynamics. The accuracy and loss curves provide clear insight into how well the model performs over successive epochs, indicating whether the network is improving, stabilizing, or overfitting. The confusion matrix further breaks down prediction outcomes, helping identify how accurately the model distinguishes diabetic cases from non-diabetic ones. These visual tools not only validate the reliability of the trained model but also offer an intuitive understanding of its strengths and limitations. Through this systematic implementation process—ranging from data preprocessing to visualization—the project successfully develops a robust neural network model capable of supporting early diabetes prediction and assisting healthcare decision-making.

6. SYSTEM ARCHITECTURE



Input

This stage loads the Pima Indians Banknote Authentication (CSV). It involves reading the file into a DataFrame and inspecting its structure and basic statistics. Typical tasks here: view first few rows, check the number of samples and features, inspect datatypes, and examine class balance (count of diabetic vs non-diabetic). This step ensures you know what variables are available (pregnancies, glucose, blood pressure, skin thickness, insulin, BMI, diabetes pedigree function, age, Outcome) and whether the dataset needs cleaning.

Preprocessing

Preprocessing prepares raw data for the ANN so the model can learn effectively and generalize well.

- Handle missing/invalid values: identify zeros, NaNs, or unrealistic values (e.g., zero BMI or glucose) and decide on a strategy — remove rows, replace with median/mean, or use domain-driven imputation.
- Feature selection / engineering (optional): remove redundant columns, create derived features (e.g., age groups, BMI categories) if useful.
- Standardize features: apply StandardScaler (zero mean, unit variance) so features with different scales (glucose vs age) don't dominate learning.
- Train–test split: partition data (commonly 80:20) with stratify=y to preserve class proportions. Optionally create a validation split or use k-fold CV.

- Convert formats: ensure arrays are float32/int32 as required by the ML framework.

Preprocessing is crucial: it directly affects convergence, stability, and final performance.

ANN Model Construction

This stage defines the neural network architecture and compilation details.

- Input layer: sized to the number of features (here, 8).
- Hidden layers: e.g., Dense(64, ReLU) → Dropout(0.2) → Dense(32, ReLU). These layers learn nonlinear feature interactions; ReLU helps with gradient flow and sparsity.
- Dropout: randomly disables a fraction of neurons during training to reduce overfitting and improve generalization.
- Output layer: Dense(1, sigmoid) — produces a probability for the positive class (diabetic).
- Compile settings: choose optimizer (Adam), loss (binary_crossentropy for two-class problems), and metrics (accuracy; optionally precision, recall, AUC). Choosing hyperparameters (layer sizes, dropout rate, learning rate) is part of architecture design and may be tuned.

The goal here is to build a model expressive enough to capture patterns but regularized enough to avoid overfitting.

Training

Training is where the network learns by updating weights to minimize loss.

- Fit the model: run for a fixed number of epochs (e.g., 35) with a chosen batch size (e.g., 32), and optionally a validation_split (e.g., 0.2) to monitor validation metrics each epoch.
- Monitor: record training & validation loss and accuracy (history object). Watch for overfitting (training accuracy rising while validation accuracy plateaus or drops).

- Callbacks (optional): use EarlyStopping to stop when validation loss stops improving, ModelCheckpoint to save best weights, and ReduceLROnPlateau to lower learning rate on plateau.
- Hyperparameter tuning: you may iterate over epochs, batch size, learning rate, layer sizes, and regularization to improve performance.

Training converts initialized weights into a predictive model by repeated forward/backward passes on the data.

Visualization and Prediction

This final stage interprets the trained model and uses it for inference.

- Visualizations:
    - *Accuracy vs Epochs* — shows learning curve for train and validation sets.
    - *Loss vs Epochs* — shows how loss decreases and can indicate over/underfitting.
    - *Confusion matrix* — shows true positives, true negatives, false positives, false negatives to understand error types.
    - *Classification report* — precision, recall, F1-score per class.
    - *Optional:* ROC curve, AUC, precision–recall curve for threshold-insensitive evaluation.
- Prediction: apply model to test set or real user inputs. Convert sigmoid outputs to class labels using a threshold (commonly 0.5), or use calibrated probabilities if required. Provide result as "Diabetic / Non-Diabetic" and optionally include the probability/confidence for each prediction.
- Interpretation & deployment: use the visual and numeric outputs to assess readiness for deployment. If acceptable, export model (e.g., model.save()), build a prediction API or a simple GUI, and document limitations (dataset bias, clinical validation requirement).

7. CODE IMPLEMENTATION

Algorithm: Diabetes Prediction using Artificial Neural Network

Input: Pima Indians Diabetes Dataset

Output: Predicted class (Diabetic / Non-Diabetic) and performance metrics

1. Start

2. Load Dataset

   2.1 Load the Pima Indians Diabetes dataset from the CSV file.

   2.2 Separate the dataset into:

   • Feature matrix X(all columns except *Outcome*)

   • Target vector y(the *Outcome* column: 0/1)

3. Preprocess Data

   3.1 Convert $X$to float32 and $y$to int32.

   3.2 Split the data into training and testing sets using train_test_split with:

   • test_size = 0.2

   • stratify = y

   3.3 Fit StandardScaler on training data X_{train}.

   3.4 Transform $X_{train}$and X_{test}using the fitted scaler.

4. Build ANN Model

   4.1 Initialize a Sequential model.

   4.2 Add input layer with shape = number of features.

   4.3 Add first hidden layer: Dense(64) with ReLU activation.

   4.4 Add Dropout layer with rate 0.2 to reduce overfitting.

   4.5 Add second hidden layer: Dense(32) with ReLU activation.

   4.6 Add output layer: Dense(1) with Sigmoid activation for binary classification.

5. Compile Model

   5.1 Set optimizer = Adam.

   5.2 Set loss function = Binary Cross-Entropy.

   5.3 Set evaluation metric = Accuracy.

6. Train Model

   6.1 Train the model on $X_{train}, y_{train}$with:

- Epochs = 35
- Batch size = 32
- Validation split = 0.2

6.2 Store training history (accuracy and loss for train and validation).

7. Test Model

7.1 Use the trained model to predict probabilities for X_{test}.

7.2 Convert probabilities to class labels:

If probability > 0.5 → predict 1 (Diabetic)

Else → predict 0 (Non-Diabetic)

8. Evaluate Performance

8.1 Compute test accuracy using accuracy_score(y_test, y_pred).

8.2 Generate classification report (precision, recall, F1-score).

8.3 Compute confusion matrix.

9. Visualize Results

9.1 Plot training vs. validation accuracy across epochs.

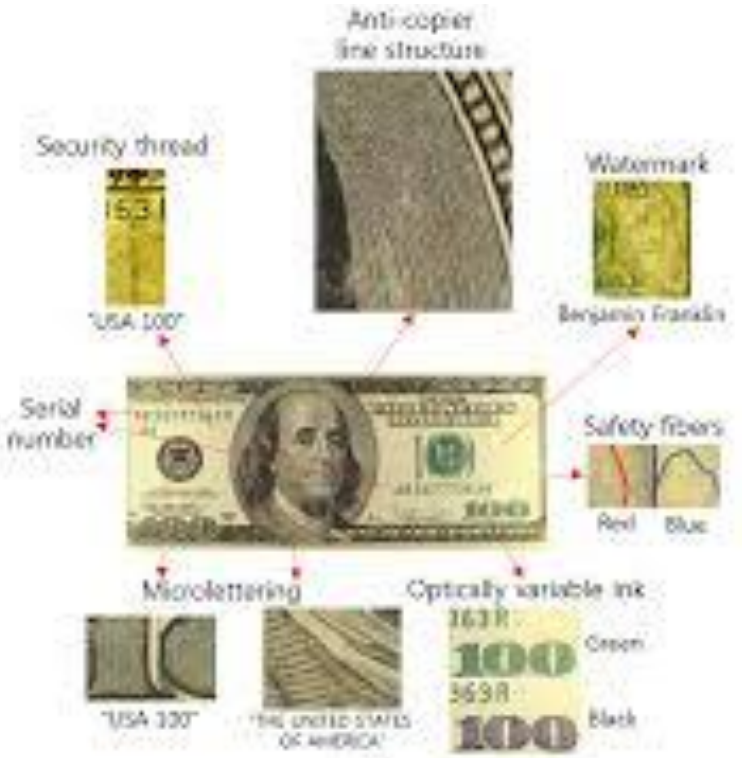
9.2 Plot training vs. validation loss across epochs.

9.3 Plot confusion matrix as a heatmap.

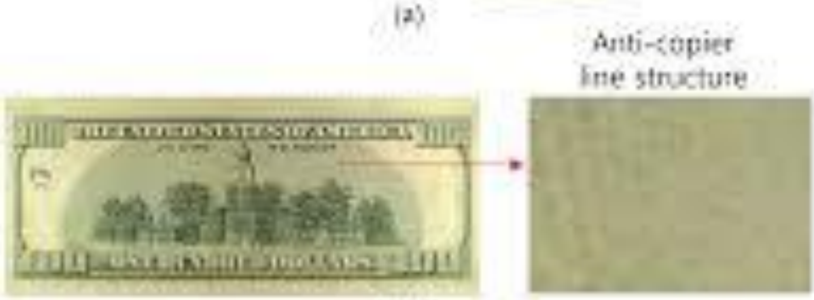
10. End

8.RESULT

(a) (b)



(a)



(b)

9. CONCLUSION

The project *"Banknote Authentication Using ANN Model"* successfully demonstrates how Artificial Neural Networks can be used as an effective and reliable tool for detecting counterfeit currency. By utilizing statistical features such as variance, skewness, kurtosis, and entropy extracted from banknote images, the ANN model was able to learn the underlying patterns that distinguish genuine notes from counterfeit ones. The experimental results indicate that the model achieves high accuracy and strong generalization performance, proving its capability to classify banknotes with minimal human intervention.

This automated approach significantly reduces the dependency on manual verification methods, which are prone to errors and require specialized skills or equipment. The ANN-based system ensures faster, more consistent, and more scalable authentication, making it suitable for real-time applications in banks, ATMs, retail sectors, and currency-processing machines. Moreover, the flexibility of neural networks allows the model to be retrained and improved as new counterfeit techniques emerge, making it a future-ready solution.

Overall, the project not only strengthens financial security by minimizing the circulation of counterfeit currency but also highlights the potential of machine learning in solving real-world security challenges. The successful implementation of this ANN-based banknote authentication system demonstrates how intelligent algorithms can enhance accuracy, efficiency, and trust in financial transactions, ultimately contributing to a more secure and reliable monetary ecosystem.

.

10. REFERENCES

[1] Ganie et al. (2023). Ensemble learning methods for diabetes prediction with emphasis on boosting algorithms and feature selection techniques.

[2] Gündoğdu (2023). Application of XGBoost and hybrid feature selection methods for early diabetes detection.

[3] Chang et al. (2023). Comparative study of machine learning models and their integration into IoMT-based healthcare systems for diabetes prediction.

[4] Tasin et al. (2022). Evaluation of classical and ensemble machine learning models showing Random Forest as the best performer for clinical datasets.

[5] Madan et al. (2022). Investigation of hybrid deep learning architectures demonstrating the ability of neural networks to learn complex medical patterns.

[6] Ayat (2024). Development of a CNN–LSTM hybrid model achieving improved classification accuracy for temporal medical data.

[7] R. Kumar & S. Verma (2022). Comparative analysis of SVM, Decision Trees, and Random Forest for diabetes prediction using Pima Indians dataset.

[8] Kaggle. (2024). Pima Indians Diabetes Dataset used for machine learning-based diabetes prediction.

[9] TensorFlow Developers. (2015–2024). TensorFlow deep learning framework used to implement ANN models.

[10] Scikit-Learn Developers. (2011–2024). Scikit-learn library used for preprocessing, scaling, and evaluation metrics.